What is a Computation ?

Prof. S.K.Laskmana Rao
BANGALORE

Man has been computing for centuries but it is no early task to answer the question: " What is a computation?". An answer emerged only in 1936 after Alan Turing's cogent and complete logical analysis of the notion of computation. Turing's analysis shows the possibility of construction of "Universal" computers that are programmed to carry out any possible computation. An offshoot of this analysis is the recognition that certain mathematical problems are incapable of coputational solution. Turing gave some simple examples of unsolvable problems and it became clear later that many mathematical problems are indeed computationally unsolvable. A precise definition of computation is based on an analysis of what a human does when he computers. He has a set of rules to be carried out in a mechanical way with the use of ingenuity in the setting up of rules to promote efficient coputation. Man employs symbles and his behaviour changes when he notes specific symbols during the computational steps. Turing saw that the essential steps in the process of calculation were simple basic operations performed repetitively many times. A product like.

1887 X 32
3774
56610
60384

needs a two – dimensional street but can as well be represented in a linear medium, viz.

1|8|8|7|X|3|2|=|3|7|7|4|+|5|6|6|1|0|=|6|0|3|8|4|

we may then adopt the conventions:

(a) Calculation on a linear medium, say a magnetic tape

(b) The only symbols to be employed are zero (0) and one (1)

(c) At a glance one skans only one symbol in a single square cell.

A calculation involves the replacement of a zero by one or a one by a zero or there may be a shift in his attention from one square cell to another which is immediately next to the right or the left of the suqare skanned. The shift is contingent on the decision of the

calculation after observation of the symbol in the square being skanned presently. There are such steps in succession and when he halts, the indication is the termination of the computation.

A computation by a human comprises work on strings of zeros and ones spread aslinear tape and the instructions carried out are of the form.

Table – 1

(a) Write the symbol 1

(b) Write the symbol 0

(c) Move one square to the right

(d) Move one square to the Left

(e) Observe the symbol skanned currently and them decide the next step

(f) Stop.

These in structions constitute a special programming Lauguage.

Table – 2

Turing – Post programming Lauguage

1	PRINT 1
2	PRINT 0
3	GORIGHT
4	GO LEFT
5	GO TO STEP (i) IF 1 IS SKANNED
6	GO TO STEP (i) IF 0 IS SKANNED
7	STOP

Each instruction in a Turimg – Post program is one of the above seven statements. For starting the calculation one must have "input" data and the program begins with the skanning of a specific square on the tape which has on it a string of zeros and ones.

An example

Table - 3

Doubling Progam

1	PRINT 0
2	GO LEFT
3	GO TO STEP 2 IF 1 IS SKANNED
4	PRINT 1
5	GO RIGHT
6	GO TO STEP 5 IF 1 IS SKANNED
7	PRINT 1
8	GO RIGHT
9	GO TO STEP 1 IF 1 IS SKANNED
10	STEP

The program has ten instructions and an initial input which is a string of zeros and ones and the computation is completely determined by the initial arrangement of symbols on the tape as also the specification of the square cell being skanned in the beginning.

The execution of the doubling program is illustreated below through an example.

Table - 4

0	0	1	1	0	0
		↑			
0	0	0	1	0	0
		↑			
0	0	0	1	0	0
	↑				
0	1	0	1	0	0
	↑				
0	1	0	1	0	0
		↑			
0	1	1	1	0	0
		↑			
0	1	1	1	0	0
			↑		
0	1	1	0	0	0
			↑		
0	1	1	0	0	0
		↑			
0	1	1	0	0	0
	↑				
0	1	1	0	0	0
↑					
1	1	1	0	0	0
↑					
1	1	1	0	0	0
	↑				
1	1	1	0	0	0
		↑			
1	1	1	0	0	0
			↑		
1	1	1	1	0	0
			↑		
1	1	1	1	0	0
				↑	

If one begins with a tape configuration the non blank portion of which consisits of a row of ones with the skanned square containing the leftmost of the ones, the doubling program halts evertually, leaving a block of twice as many ones on the tape as were there to begin with. On executing the STOP instruction, the computation halts and the computation goes on forver if the STOP instruction is not encountered in the course of ture computation. An important questio is to know when we can say that a computation will course to a halt eventually.

An example. Consider the three step Turing – Post program

Table – 5

1	GO RIGHT
2	GO TO STEP 1 IF 0 IS SKANNED
3	STOP

This program halts as soon as the motion to the right reaches as square containing the symbol 1.

Table – 6

0	0	0	1	1	1	0	0	0
					↑			
0	0	0	1	1	1	0	0	0
						↑		
0	0	0	1	1	1	0	0	0
							↑	

Steps 1 and 2 will be carried out again and agains if 1 is not skanned, and this forcing the message that a computation from a Turing – Post program need not halt at all.

An interesting question is whether it is possible to predict the eventual halting or other wise of a Turing – Post program.

Codes for Turing – Post programs

It is possible to encode a Turing – Post program by a string of zeros and ones and such a string can be placed on the tape and this can also be visualized as the input candidat p for the same or another Turing – Post program. We may thus think of

Turing – Post programs as being capable of performing computations on other Turing – Post programs.

Table – 7

Representation of Turing – Post instructions by a string of zeros and ones.

Code	Program Instruction
0 0 0	Print 0
0 0 1	Print 1
0 1 0	Go Left
0 1 1	Go right
1 0 1 0 .... 0 1	Go to step (i) it o is skanned
1 1 0 1 1 ..1 0	Go to step (i) is 1 is skanned
1 0 0	Step

e.g. "Go to step 3 it 0 is skanned" is represented by the code 1010001.
(3)

To represent and entire program, simply write in order the representation of each individual instruction of the program, placeing a 1 at the very beginning and a 1 1 1 at the very end as punctuation marks.

e.g. The Code for the doubling program is the string:

Table – 8

1(0 0 0) (0 1 0) (1 1 0 1 1 0) (0 0 1) (0 1 1) (1 1 0 1 1 1 1 1 0).... (0 0 1) (0 1 1) (1 1 0 1 0) (1 0 0) 1 1 1

The brackets are enclosed merely as aid to show the individual instructions in the program.

It is easy to believe that the code of any Turing – Post program can be described in a unique way by direct means to obtain the program of which it is the code.

{(Remove the invitial 1), (remove the final 1 1 1)} as these signify the marks of punctuation, proceed from left to right in the code and observe the first group of three digits. The group is one of the triples: (0 0 0), (0 0 1), (0 1 0), (0 1 1), (1 0 0). The three groups (1 0 1), (1 1 0), (111) do not enter the representation of table 7. The program

instruction correponding to the above five triples are (PRINT 0), (PRINT 1), (GO LEFT), (GO RIGHT), (STO) respectively. The code may also have the forms (1 0 1 $\underbrace{0\,0\,0\,.\,.\,0}_{(i)}$ 1) and (1 1 0 $\underbrace{1\,1\,.\,.\,.\,1}_{(i)}$ 0) and these indicate the instructions (GO TO STEP (i) IF 0 IS SKANNED) and (GO TO STEP (i) IF 1 IS SKANNED) respectively. After the first instruction, cross out its code in the string and continue trips process, always goving from left to right. This leaders to the decoding of the string.

Table – 9

Example of the doubling program

| | | |

Let us review were how a human (A) works out the calculation. After seeing the tape contents on which w is to work and skanning the string of zeros and ones from left to right, he search for the first place of appearance of 1 1 1. This triple marks the end of the code (p) and the begimming of the input string. (A) can write code (p) on a sheet of paper and the input string on anther and start decoding of the code (p) and reach the actual Turing – Post program P. He then executes the instructions of P on the input in a robot like fashion. It the computation halts, he takes the final tape contents as the output. Thus he can do what we want $\cup$ to do. Turing's analysis of the computation process suggests that there is a Turing – Post program which can carry out the above described process i.e. we have seen a universal Turing – Post program. With greater effort one can actually write in detail an explicit universal Turing – Post program – This was indeed does by Turing of in his 1936 paper -. For any particular Turing – Post program P, it is possible to build a machine that executes the intructions in P in the same sequence. Hence, this can also be done for the universal program and the mechine we get this way is an exaple of all-purpose or univrsal computing machine. The code for a prticular program P placed on the tape can be throught of as a 'program' for excuting the computation which P does. Thus the concept of an all purpose computes which can be programmed to carry out any computaion wat so ever.

The Halting problem

The halting problem for a partcular Turning-Post program is the problem of identifying those initial tape configurations that enable the program to halt eventually, as against those for which the computation never halts. certain input strings may cause a program to run forever, by an infinite loop that msy be caused by the institution GO TO, can one computationally test a given tape configuration to see whether or not the program eventually halts, after begning with that tape configuration ?

There is no computational procedure for testing a given tape expression to find out if the univerrsal program U halts evntually aften begning with that tapeconfiguration.

This is an instance of an easily stated and understood question, that has no solution.

Example: (1) Trisection of an asngle θ using only the implements ; astraight edge and a pair of compasses.

The process halts here for special inputs like $\theta = \frac{\pi}{2^{n+1}}, \frac{\pi}{(5)(2^n)}$ (n=0,1,2,…)

The infinite sum $\frac{1}{4}+\frac{1}{4^2}+\frac{1}{4^3}+\ldots\ldots\ldots=\frac{1}{3}$

shows that there is a limit operation and thus an infinite loop in the computation process.

(2) Integer solutions of the Diophsntine equation ; 6x-3y =20.

No solution exists.

David Hilbert(1862-1943) gave a list of 23 important problems at the International conference of mathematicians (1900) for the attention of the mathematical community. The 10th problem in the lst is to find a computing procedure for testing a Diophantine equation and say if it has or has not any integer solution (r). In 1960, Martin Davis, Hilary Putnam and Julia Robinson proved that if there is one Diophantine equation whose solutions satisfy a special condition involving the relative size of the numbers constituting such a solution ,then Hilbert problem is unsolvable. Later 1970, Yuri Mathiyasevich (then a lad of 22) clinched the equation by actually proving the unsolvability of Hilbert's 10th problem.

In 1920, David Hilbert posed the following problem in mathematical logic.Given a finite list of statements (called premises)and an additional statement (called conclusion) and use of six operations

(i) not (~), (ii) and (∩), (iii) or (∪) (iv) implies (⇒), (v) for all (∀), (vi) there exist (∃),the problem was to devise or compute a procedure for testing whether or not the conclusion can be deduced from the premises ,using the rules of logic.the existence of such a procedure is out of question. Like the halting problem, this again is an instance of unsolvable mathematical problems.

undecidable statements:

Through their magnumopus, (Principia Mathematica), Betrand Russel and Alfred North whitended made it clear(1920) that all existing proofs in mathematics could be translated into speicfic logicalsystems provided in their siminal work. The view prevailed for about a decade that their work would suffice to prove or disprove any matematical statements. In 1931, Kurt Gödel showed that there are statements about integers which can neither be proved or disproved in the logical system of Russel and Whitehead. Such statements are undessidable.

Computation is often carried out to obtain a desired answer. We have seen the distinctins between a computation that halts eventually and one which never halts. It is natural to ask how complex need a Turning-post program be, to produce an output for a given input.this question is intimately connectedto the mathematical theory of randomness and extensions of Gödel's work on undecidability.

- o O o -